**Practical 11**

**import tensorflow as tf**

**from tensorflow.keras import datasets, layers, models**

**import matplotlib.pyplot as plt**

**# Load and preprocess the CIFAR10 dataset**

**(train\_images, train\_labels), (test\_images, test\_labels) = datasets.cifar10.load\_data()**

**train\_images, test\_images = train\_images / 255.0, test\_images / 255.0**

**# Define the CNN model**

**model = models.Sequential()**

**model.add(layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(32, 32, 3)))**

**model.add(layers.MaxPooling2D((2, 2)))**

**model.add(layers.Conv2D(64, (3, 3), activation='relu'))**

**model.add(layers.MaxPooling2D((2, 2)))**

**model.add(layers.Conv2D(64, (3, 3), activation='relu'))**

**model.add(layers.Flatten())**

**model.add(layers.Dense(64, activation='relu'))**

**model.add(layers.Dense(10))**

**# Compile the model**

**model.compile(optimizer='adam',**

**loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True),**

**metrics=['accuracy'])**

**# Train the model**

**history = model.fit(train\_images, train\_labels, epochs=10,**

**validation\_data=(test\_images, test\_labels))**

**# Evaluate the model**

**test\_loss, test\_acc = model.evaluate(test\_images,  test\_labels, verbose=2)**

**print(f"Test accuracy: {test\_acc}")**

**# Plot training history (optional)**

**plt.plot(history.history['accuracy'], label='accuracy')**

**plt.plot(history.history['val\_accuracy'], label = 'val\_accuracy')**

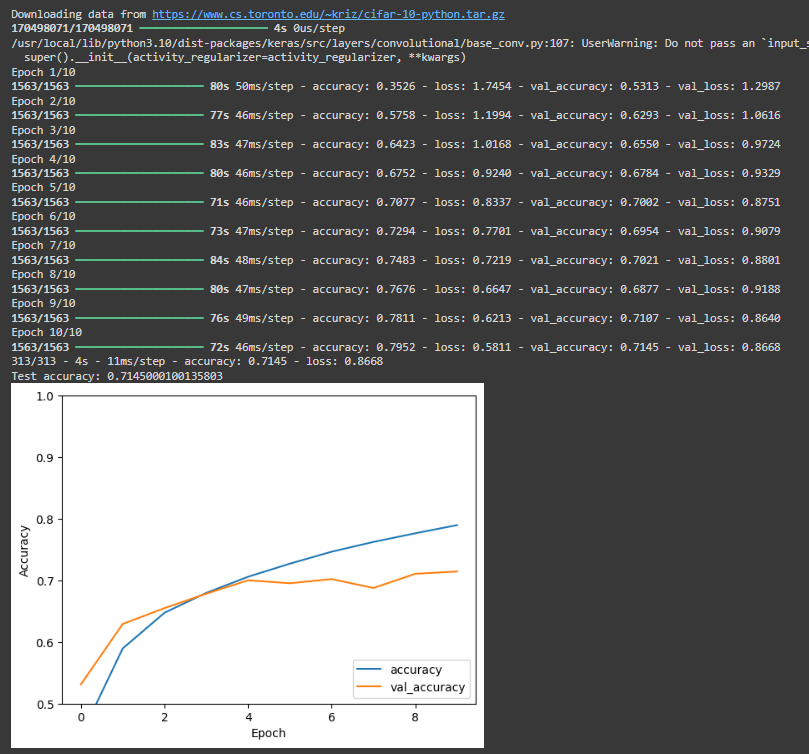
**plt.xlabel('Epoch')**

**plt.ylabel('Accuracy')**

**plt.ylim([0.5, 1])**

**plt.legend(loc='lower right')**

**plt.show()**

****

**Object detection using CNN**

**import tensorflow as tf**

**import numpy as np**

**import cv2**

**from tensorflow.keras.losses import mse**

**# Load the saved model**

**model = tf.keras.models.load\_model('my\_cifar10\_model.h5', custom\_objects={'mse': mse})**

**# Define class names for CIFAR-10**

**class\_names = ['airplane', 'automobile', 'bird', 'cat', 'deer',**

**'dog', 'frog', 'horse', 'ship', 'truck']**

**# Display the uploaded image**

**plt.imshow(img)**

**plt.axis('off')**

**plt.show()**

**# Load and preprocess an image (replace with your own image)**

**img\_path = '/content/download.jpg'  # Replace with the path to your image**

**img = cv2.imread(img\_path)**

**img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)**

**img = cv2.resize(img, (32, 32))  # Resize to CIFAR-10 input size**

**img = img / 255.0  # Normalize**

**# Make a prediction**

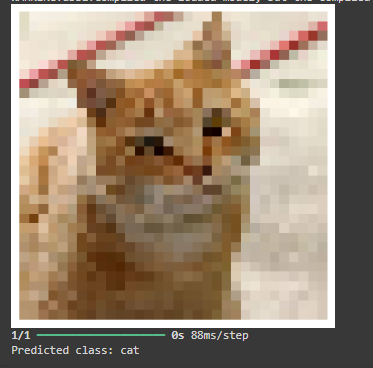
**img\_array = np.expand\_dims(img, axis=0)**

**predictions = model.predict(img\_array)**

**predicted\_class = np.argmax(predictions[0])**

**# Display the result**

**print(f"Predicted class: {class\_names[predicted\_class]}")**

****